

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

[PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2005-08/6384.html>

From: Chris Wright (chrisw_at_osdl.org)

Date: 08/25/05

Date: Wed, 24 Aug 2005 18:20:31 -0700

To: linux-security-module@wirex.com

Call security hooks conditionally if the security_op is filled out.
Branches can be more efficient than the unconditional indirect function call. Inspired by patch from Kurt Garloff <garloff@suse.de>.

Signed-off-by: Chris Wright <chrisw@osdl.org>

```
---
include/linux/security.h | 825 ++++++-----
1 files changed, 411 insertions(+), 414 deletions(-)
Index: linux-2.6/include/linux/security.h
=====
--- linux-2.6.orig/include/linux/security.h
+++ linux-2.6/include/linux/security.h
@@ -1264,10 +1264,10 @@ static inline int security_init(void)
static inline int security_ptrace (struct task_struct * parent, struct task_struct * child)
{
#ifdef CONFIG_SECURITY
-   return security_ops->ptrace (parent, child);
-#else
-   return cap_ptrace (parent, child);
+   if (security_ops->ptrace)
+       return security_ops->ptrace(parent, child);
#endif
+   return cap_ptrace (parent, child);
}

@@ -1277,10 +1277,10 @@ static inline int security_capget (struct
                                kernel_cap_t *permitted)
{
#ifdef CONFIG_SECURITY
-   return security_ops->capget (target, effective, inheritable, permitted);
-#else
-   return cap_capget (target, effective, inheritable, permitted);
+   if (security_ops->capget)
+       return security_ops->capget(target, effective, inheritable, permitted);
#endif
+   return cap_capget (target, effective, inheritable, permitted);
}

static inline int security_capset_check (struct task_struct *target,
@@ -1289,10 +1289,10 @@ static inline int security_capset_check
```

[PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```

                                kernel_cap_t *permitted)
{
#ifdef CONFIG_SECURITY
-   return security_ops->capset_check (target, effective, inheritable, permitted);
-#else
+   if (security_ops->capset_check)
+       return security_ops->capset_check(target, effective, inheritable, permitted);
+#endif
    return cap_capset_check (target, effective, inheritable, permitted);
-#endif
}

static inline void security_capset_set (struct task_struct *target,
@@ -1301,183 +1301,183 @@ static inline void security_capset_set (
                                kernel_cap_t *permitted)
{
#ifdef CONFIG_SECURITY
-   security_ops->capset_set (target, effective, inheritable, permitted);
-#else
-   cap_capset_set (target, effective, inheritable, permitted);
+   if (security_ops->capset_set)
+       return security_ops->capset_set(target, effective, inheritable, permitted);
+#endif
+   cap_capset_set (target, effective, inheritable, permitted);
}

static inline int security_acct (struct file *file)
{
#ifdef CONFIG_SECURITY
-   return security_ops->acct (file);
-#else
-   return 0;
+   if (security_ops->acct)
+       return security_ops->acct(file);
+#endif
+   return 0;
}

static inline int security_sysctl(struct ctl_table *table, int op)
{
#ifdef CONFIG_SECURITY
-   return security_ops->sysctl(table, op);
-#else
-   return 0;
+   if (security_ops->sysctl)
+       return security_ops->sysctl(table, op);
+#endif
+   return 0;
}

static inline int security_quotactl (int cmds, int type, int id,
                                struct super_block *sb)
{
#ifdef CONFIG_SECURITY
-   return security_ops->quotactl (cmds, type, id, sb);
-#else
-   return 0;
+   if (security_ops->quotactl)
+       return security_ops->quotactl(cmds, type, id, sb);
+#endif
+   return 0;
}

```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
static inline int security_quota_on (struct dentry * dentry)
{
#ifdef CONFIG_SECURITY
-   return security_ops->quota_on (dentry);
-#else
-   return 0;
+   if (security_ops->quota_on)
+       return security_ops->quota_on(dentry);
#endif
+   return 0;
}

static inline int security_syslog(int type)
{
#ifdef CONFIG_SECURITY
-   return security_ops->syslog(type);
-#else
-   return cap_syslog(type);
+   if (security_ops->syslog)
+       return security_ops->syslog(type);
#endif
+   return cap_syslog(type);
}

static inline int security_settime(struct timespec *ts, struct timezone *tz)
{
#ifdef CONFIG_SECURITY
-   return security_ops->settime(ts, tz);
-#else
-   return cap_settime(ts, tz);
+   if (security_ops->settime)
+       return security_ops->settime(ts, tz);
#endif
+   return cap_settime(ts, tz);
}

static inline int security_vm_enough_memory(long pages)
{
#ifdef CONFIG_SECURITY
-   return security_ops->vm_enough_memory(pages);
-#else
-   return cap_vm_enough_memory(pages);
+   if (security_ops->vm_enough_memory)
+       return security_ops->vm_enough_memory(pages);
#endif
+   return cap_vm_enough_memory(pages);
}

static inline int security_bprm_alloc (struct linux_binprm *bprm)
{
#ifdef CONFIG_SECURITY
-   return security_ops->bprm_alloc_security (bprm);
-#else
-   return 0;
+   if (security_ops->bprm_alloc_security)
+       return security_ops->bprm_alloc_security(bprm);
#endif
+   return 0;
}

static inline void security_bprm_free (struct linux_binprm *bprm)
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
{
#ifdef CONFIG_SECURITY
-   security_ops->bprm_free_security (bprm);
-#else
-   return;
+   if (security_ops->bprm_free_security)
+       return security_ops->bprm_free_security(bprm);
#endif
+   return;
}

static inline void security_bprm_apply_creds (struct linux_binprm *bprm, int unsafe)
{
#ifdef CONFIG_SECURITY
-   security_ops->bprm_apply_creds (bprm, unsafe);
-#else
-   cap_bprm_apply_creds (bprm, unsafe);
+   if (security_ops->bprm_apply_creds)
+       return security_ops->bprm_apply_creds(bprm, unsafe);
#endif
+   cap_bprm_apply_creds (bprm, unsafe);
}

static inline void security_bprm_post_apply_creds (struct linux_binprm *bprm)
{
#ifdef CONFIG_SECURITY
-   security_ops->bprm_post_apply_creds (bprm);
-#else
-   return;
+   if (security_ops->bprm_post_apply_creds)
+       return security_ops->bprm_post_apply_creds(bprm);
#endif
+   return;
}

static inline int security_bprm_set (struct linux_binprm *bprm)
{
#ifdef CONFIG_SECURITY
-   return security_ops->bprm_set_security (bprm);
-#else
-   return cap_bprm_set_security (bprm);
+   if (security_ops->bprm_set_security)
+       return security_ops->bprm_set_security(bprm);
#endif
+   return cap_bprm_set_security (bprm);
}

static inline int security_bprm_check (struct linux_binprm *bprm)
{
#ifdef CONFIG_SECURITY
-   return security_ops->bprm_check_security (bprm);
-#else
-   return 0;
+   if (security_ops->bprm_check_security)
+       return security_ops->bprm_check_security(bprm);
#endif
+   return 0;
}

static inline int security_bprm_secureexec (struct linux_binprm *bprm)
{
#ifdef CONFIG_SECURITY
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
-     return security_ops->bprm_secureexec (bprm);
-#else
-     return cap_bprm_secureexec(bprm);
+     if (security_ops->bprm_secureexec)
+         return security_ops->bprm_secureexec(bprm);
+ #endif
+     return cap_bprm_secureexec(bprm);
}

static inline int security_sb_alloc (struct super_block *sb)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_alloc_security (sb);
-#else
-     return 0;
+     if (security_ops->sb_alloc_security)
+         return security_ops->sb_alloc_security(sb);
+ #endif
+     return 0;
}

static inline void security_sb_free (struct super_block *sb)
{
#ifdef CONFIG_SECURITY
-     security_ops->sb_free_security (sb);
-#else
-     return;
+     if (security_ops->sb_free_security)
+         return security_ops->sb_free_security(sb);
+ #endif
+     return;
}

static inline int security_sb_copy_data (struct file_system_type *type,
                                       void *orig, void *copy)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_copy_data (type, orig, copy);
-#else
-     return 0;
+     if (security_ops->sb_copy_data)
+         return security_ops->sb_copy_data(type, orig, copy);
+ #endif
+     return 0;
}

static inline int security_sb_kern_mount (struct super_block *sb, void *data)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_kern_mount (sb, data);
-#else
-     return 0;
+     if (security_ops->sb_kern_mount)
+         return security_ops->sb_kern_mount(sb, data);
+ #endif
+     return 0;
}

static inline int security_sb_statfs (struct super_block *sb)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_statfs (sb);
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
-#else
-     return 0;
+     if (security_ops->sb_statfs)
+         return security_ops->sb_statfs(sb);
#endif
+     return 0;
}

static inline int security_sb_mount (char *dev_name, struct nameidata *nd,
@@ -1485,96 +1485,96 @@ static inline int security_sb_mount (cha
                                void *data)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_mount (dev_name, nd, type, flags, data);
-#else
-     return 0;
+     if (security_ops->sb_mount)
+         return security_ops->sb_mount(dev_name, nd, type, flags, data);
#endif
+     return 0;
}

static inline int security_sb_check_sb (struct vfsmount *mnt,
                                struct nameidata *nd)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_check_sb (mnt, nd);
-#else
-     return 0;
+     if (security_ops->sb_check_sb)
+         return security_ops->sb_check_sb(mnt, nd);
#endif
+     return 0;
}

static inline int security_sb_umount (struct vfsmount *mnt, int flags)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_umount (mnt, flags);
-#else
-     return 0;
+     if (security_ops->sb_umount)
+         return security_ops->sb_umount(mnt, flags);
#endif
+     return 0;
}

static inline void security_sb_umount_close (struct vfsmount *mnt)
{
#ifdef CONFIG_SECURITY
-     security_ops->sb_umount_close (mnt);
-#else
-     return;
+     if (security_ops->sb_umount_close)
+         return security_ops->sb_umount_close(mnt);
#endif
+     return;
}

static inline void security_sb_umount_busy (struct vfsmount *mnt)
{
#ifdef CONFIG_SECURITY
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
-     security_ops->sb_umount_busy (mnt);
-#else
-     return;
+     if (security_ops->sb_umount_busy)
+         return security_ops->sb_umount_busy(mnt);
+endif
+     return;
}

static inline void security_sb_post_remount (struct vfsmount *mnt,
                                           unsigned long flags, void *data)
{
#ifdef CONFIG_SECURITY
-     security_ops->sb_post_remount (mnt, flags, data);
-#else
-     return;
+     if (security_ops->sb_post_remount)
+         return security_ops->sb_post_remount(mnt, flags, data);
+endif
+     return;
}

static inline void security_sb_post_mountroot (void)
{
#ifdef CONFIG_SECURITY
-     security_ops->sb_post_mountroot ();
-#else
-     return;
+     if (security_ops->sb_post_mountroot)
+         return security_ops->sb_post_mountroot();
+endif
+     return;
}

static inline void security_sb_post_addmount (struct vfsmount *mnt,
                                           struct nameidata *mountpoint_nd)
{
#ifdef CONFIG_SECURITY
-     security_ops->sb_post_addmount (mnt, mountpoint_nd);
-#else
-     return;
+     if (security_ops->sb_post_addmount)
+         return security_ops->sb_post_addmount(mnt, mountpoint_nd);
+endif
+     return;
}

static inline int security_sb_pivotroot (struct nameidata *old_nd,
                                       struct nameidata *new_nd)
{
#ifdef CONFIG_SECURITY
-     return security_ops->sb_pivotroot (old_nd, new_nd);
-#else
-     return 0;
+     if (security_ops->sb_pivotroot)
+         return security_ops->sb_pivotroot(old_nd, new_nd);
+endif
+     return 0;
}

static inline void security_sb_post_pivotroot (struct nameidata *old_nd,
                                           struct nameidata *new_nd)
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
{
#ifdef CONFIG_SECURITY
-   security_ops->sb_post_pivotroot (old_nd, new_nd);
-#else
-   return;
+   if (security_ops->sb_post_pivotroot)
+       return security_ops->sb_post_pivotroot(old_nd, new_nd);
#endif
+   return;
}

static inline int security_inode_alloc (struct inode *inode)
@@ -1582,10 +1582,10 @@ static inline int security_inode_alloc (
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (inode)))
        return 0;
-   return security_ops->inode_alloc_security (inode);
-#else
-   return 0;
+   if (security_ops->inode_alloc_security)
+       return security_ops->inode_alloc_security(inode);
#endif
+   return 0;
}

static inline void security_inode_free (struct inode *inode)
@@ -1593,10 +1593,10 @@ static inline void security_inode_free (
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (inode)))
        return;
-   security_ops->inode_free_security (inode);
-#else
-   return;
+   if (security_ops->inode_free_security)
+       return security_ops->inode_free_security(inode);
#endif
+   return;
}

static inline int security_inode_create (struct inode *dir,
@@ -1606,10 +1606,10 @@ static inline int security_inode_create
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dir)))
        return 0;
-   return security_ops->inode_create (dir, dentry, mode);
-#else
-   return 0;
+   if (security_ops->inode_create)
+       return security_ops->inode_create(dir, dentry, mode);
#endif
+   return 0;
}

static inline void security_inode_post_create (struct inode *dir,
@@ -1619,10 +1619,10 @@ static inline void security_inode_post_c
#ifdef CONFIG_SECURITY
    if (dentry->d_inode && unlikely (IS_PRIVATE (dentry->d_inode)))
        return;
-   security_ops->inode_post_create (dir, dentry, mode);
-#else
-   return;
+   if (security_ops->inode_post_create)
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+         return security_ops->inode_post_create(dir, dentry, mode);
+     #endif
+     return;
+ }

static inline int security_inode_link (struct dentry *old_dentry,
@@ -1632,10 +1632,10 @@ static inline int security_inode_link (s
+ #ifdef CONFIG_SECURITY
+     if (unlikely (IS_PRIVATE (old_dentry->d_inode)))
+         return 0;
-     return security_ops->inode_link (old_dentry, dir, new_dentry);
-#else
-     return 0;
+     if (security_ops->inode_link)
+         return security_ops->inode_link(old_dentry, dir, new_dentry);
+ #endif
+     return 0;
+ }

static inline void security_inode_post_link (struct dentry *old_dentry,
@@ -1645,10 +1645,10 @@ static inline void security_inode_post_l
+ #ifdef CONFIG_SECURITY
+     if (new_dentry->d_inode && unlikely (IS_PRIVATE (new_dentry->d_inode)))
+         return;
-     security_ops->inode_post_link (old_dentry, dir, new_dentry);
-#else
-     return;
+     if (security_ops->inode_post_link)
+         return security_ops->inode_post_link(old_dentry, dir, new_dentry);
+ #endif
+     return;
+ }

static inline int security_inode_unlink (struct inode *dir,
@@ -1657,10 +1657,10 @@ static inline int security_inode_unlink
+ #ifdef CONFIG_SECURITY
+     if (unlikely (IS_PRIVATE (dentry->d_inode)))
+         return 0;
-     return security_ops->inode_unlink (dir, dentry);
-#else
-     return 0;
+     if (security_ops->inode_unlink)
+         return security_ops->inode_unlink(dir, dentry);
+ #endif
+     return 0;
+ }

static inline int security_inode_symlink (struct inode *dir,
@@ -1670,10 +1670,10 @@ static inline int security_inode_symlink
+ #ifdef CONFIG_SECURITY
+     if (unlikely (IS_PRIVATE (dir)))
+         return 0;
-     return security_ops->inode_symlink (dir, dentry, old_name);
-#else
-     return 0;
+     if (security_ops->inode_symlink)
+         return security_ops->inode_symlink(dir, dentry, old_name);
+ #endif
+     return 0;
+ }

static inline void security_inode_post_symlink (struct inode *dir,
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
@@ -1683,10 +1683,10 @@ static inline void security_inode_post_s
#ifdef CONFIG_SECURITY
    if (dentry->d_inode && unlikely (IS_PRIVATE (dentry->d_inode)))
        return;
-    security_ops->inode_post_symlink (dir, dentry, old_name);
-#else
-    return;
+    if (security_ops->inode_post_symlink)
+        return security_ops->inode_post_symlink(dir, dentry, old_name);
#endif
+    return;
}

static inline int security_inode_mkdir (struct inode *dir,
@@ -1696,10 +1696,10 @@ static inline int security_inode_mkdir (
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dir)))
        return 0;
-    return security_ops->inode_mkdir (dir, dentry, mode);
-#else
-    return 0;
+    if (security_ops->inode_mkdir)
+        return security_ops->inode_mkdir(dir, dentry, mode);
#endif
+    return 0;
}

static inline void security_inode_post_mkdir (struct inode *dir,
@@ -1709,10 +1709,10 @@ static inline void security_inode_post_m
#ifdef CONFIG_SECURITY
    if (dentry->d_inode && unlikely (IS_PRIVATE (dentry->d_inode)))
        return;
-    security_ops->inode_post_mkdir (dir, dentry, mode);
-#else
-    return;
+    if (security_ops->inode_post_mkdir)
+        return security_ops->inode_post_mkdir(dir, dentry, mode);
#endif
+    return;
}

static inline int security_inode_rmdir (struct inode *dir,
@@ -1721,10 +1721,10 @@ static inline int security_inode_rmdir (
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dentry->d_inode)))
        return 0;
-    return security_ops->inode_rmdir (dir, dentry);
-#else
-    return 0;
+    if (security_ops->inode_rmdir)
+        return security_ops->inode_rmdir(dir, dentry);
#endif
+    return 0;
}

static inline int security_inode_mknod (struct inode *dir,
@@ -1734,10 +1734,10 @@ static inline int security_inode_mknod (
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dir)))
        return 0;
-    return security_ops->inode_mknod (dir, dentry, mode, dev);
-#else
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
-     return 0;
+     if (security_ops->inode_mknod)
+         return security_ops->inode_mknod(dir, dentry, mode, dev);
+ #endif
+     return 0;
+ }

static inline void security_inode_post_mknod (struct inode *dir,
@@ -1747,10 +1747,10 @@ static inline void security_inode_post_m
+ #ifdef CONFIG_SECURITY
+     if (dentry->d_inode && unlikely (IS_PRIVATE (dentry->d_inode)))
+         return;
-     security_ops->inode_post_mknod (dir, dentry, mode, dev);
- #else
-     return;
+     if (security_ops->inode_post_mknod)
+         return security_ops->inode_post_mknod(dir, dentry, mode, dev);
+ #endif
+     return;
+ }

static inline int security_inode_rename (struct inode *old_dir,
@@ -1762,11 +1762,10 @@ static inline int security_inode_rename
+     if (unlikely (IS_PRIVATE (old_dentry->d_inode) ||
+         (new_dentry->d_inode && IS_PRIVATE (new_dentry->d_inode))))
+         return 0;
-     return security_ops->inode_rename (old_dir, old_dentry,
-         new_dir, new_dentry);
- #else
-     return 0;
+     if (security_ops->inode_rename)
+         return security_ops->inode_rename(old_dir, old_dentry, new_dir, new_dentry);
+ #endif
+     return 0;
+ }

static inline void security_inode_post_rename (struct inode *old_dir,
@@ -1778,11 +1777,10 @@ static inline void security_inode_post_r
+     if (unlikely (IS_PRIVATE (old_dentry->d_inode) ||
+         (new_dentry->d_inode && IS_PRIVATE (new_dentry->d_inode))))
+         return;
-     security_ops->inode_post_rename (old_dir, old_dentry,
-         new_dir, new_dentry);
- #else
-     return;
+     if (security_ops->inode_post_rename)
+         return security_ops->inode_post_rename(old_dir, old_dentry, new_dir, new_dentry);
+ #endif
+     return;
+ }

static inline int security_inode_readlink (struct dentry *dentry)
@@ -1790,10 +1788,10 @@ static inline int security_inode_readlin
+ #ifdef CONFIG_SECURITY
+     if (unlikely (IS_PRIVATE (dentry->d_inode)))
+         return 0;
-     return security_ops->inode_readlink (dentry);
- #else
-     return 0;
+     if (security_ops->inode_readlink)
+         return security_ops->inode_readlink(dentry);
+ #endif
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+     return 0;
+ }

static inline int security_inode_follow_link (struct dentry *dentry,
@@ -1802,10 +1800,10 @@ static inline int security_inode_follow_
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dentry->d_inode)))
        return 0;
-     return security_ops->inode_follow_link (dentry, nd);
-#else
-     return 0;
+     if (security_ops->inode_follow_link)
+         return security_ops->inode_follow_link(dentry, nd);
#endif
+     return 0;
+ }

static inline int security_inode_permission (struct inode *inode, int mask,
@@ -1814,10 +1812,10 @@ static inline int security_inode_permiss
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (inode)))
        return 0;
-     return security_ops->inode_permission (inode, mask, nd);
-#else
-     return 0;
+     if (security_ops->inode_permission)
+         return security_ops->inode_permission(inode, mask, nd);
#endif
+     return 0;
+ }

static inline int security_inode_setattr (struct dentry *dentry,
@@ -1826,10 +1824,10 @@ static inline int security_inode_setattr
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dentry->d_inode)))
        return 0;
-     return security_ops->inode_setattr (dentry, attr);
-#else
-     return 0;
+     if (security_ops->inode_setattr)
+         return security_ops->inode_setattr(dentry, attr);
#endif
+     return 0;
+ }

static inline int security_inode_getattr (struct vfsmount *mnt,
@@ -1838,10 +1836,10 @@ static inline int security_inode_getattr
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dentry->d_inode)))
        return 0;
-     return security_ops->inode_getattr (mnt, dentry);
-#else
-     return 0;
+     if (security_ops->inode_getattr)
+         return security_ops->inode_getattr(mnt, dentry);
#endif
+     return 0;
+ }

static inline void security_inode_delete (struct inode *inode)
@@ -1849,10 +1847,10 @@ static inline void security_inode_delete
#ifdef CONFIG_SECURITY
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
        if (unlikely (IS_PRIVATE (inode)))
            return;
-       security_ops->inode_delete (inode);
-#else
-       return;
+       if (security_ops->inode_delete)
+           return security_ops->inode_delete(inode);
    #endif
+       return;
    }

    static inline int security_inode_setxattr (struct dentry *dentry, char *name,
@@ -1861,10 +1859,10 @@ static inline int security_inode_setxatt
    #ifdef CONFIG_SECURITY
        if (unlikely (IS_PRIVATE (dentry->d_inode)))
            return 0;
-       return security_ops->inode_setxattr (dentry, name, value, size, flags);
-#else
-       return cap_inode_setxattr(dentry, name, value, size, flags);
+       if (security_ops->inode_setxattr)
+           return security_ops->inode_setxattr(dentry, name, value, size, flags);
    #endif
+       return cap_inode_setxattr(dentry, name, value, size, flags);
    }

    static inline void security_inode_post_setxattr (struct dentry *dentry, char *name,
@@ -1873,10 +1871,10 @@ static inline void security_inode_post_s
    #ifdef CONFIG_SECURITY
        if (unlikely (IS_PRIVATE (dentry->d_inode)))
            return;
-       security_ops->inode_post_setxattr (dentry, name, value, size, flags);
-#else
-       return;
+       if (security_ops->inode_post_setxattr)
+           return security_ops->inode_post_setxattr(dentry, name, value, size, flags);
    #endif
+       return;
    }

    static inline int security_inode_getxattr (struct dentry *dentry, char *name)
@@ -1884,10 +1882,10 @@ static inline int security_inode_getxatt
    #ifdef CONFIG_SECURITY
        if (unlikely (IS_PRIVATE (dentry->d_inode)))
            return 0;
-       return security_ops->inode_getxattr (dentry, name);
-#else
-       return 0;
+       if (security_ops->inode_getxattr)
+           return security_ops->inode_getxattr(dentry, name);
    #endif
+       return 0;
    }

    static inline int security_inode_listxattr (struct dentry *dentry)
@@ -1895,10 +1893,10 @@ static inline int security_inode_listxat
    #ifdef CONFIG_SECURITY
        if (unlikely (IS_PRIVATE (dentry->d_inode)))
            return 0;
-       return security_ops->inode_listxattr (dentry);
-#else
-       return 0;
+       if (security_ops->inode_listxattr)
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+         return security_ops->inode_listxattr(dentry);
+     #endif
+     return 0;
+ }

static inline int security_inode_removexattr (struct dentry *dentry, char *name)
@@ -1906,10 +1904,10 @@ static inline int security_inode_removex
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (dentry->d_inode)))
        return 0;
-     return security_ops->inode_removexattr (dentry, name);
-#else
-     return cap_inode_removexattr(dentry, name);
+     if (security_ops->inode_removexattr)
+         return security_ops->inode_removexattr(dentry, name);
+ #endif
+     return cap_inode_removexattr(dentry, name);
+ }

static inline int security_inode_getsecurity(struct inode *inode, const char *name, void *buffer)
@@ -1917,10 +1915,10 @@ static inline int security_inode_getsecu
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (inode)))
        return 0;
-     return security_ops->inode_getsecurity(inode, name, buffer, size);
-#else
-     return -EOPNOTSUPP;
+     if (security_ops->inode_getsecurity)
+         return security_ops->inode_getsecurity(inode, name, buffer, size);
+ #endif
+     return -EOPNOTSUPP;
+ }

static inline int security_inode_setsecurity(struct inode *inode, const char *name, const void *value, int size, int flags)
@@ -1928,10 +1926,10 @@ static inline int security_inode_setsecu
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (inode)))
        return 0;
-     return security_ops->inode_setsecurity(inode, name, value, size, flags);
-#else
-     return -EOPNOTSUPP;
+     if (security_ops->inode_setsecurity)
+         return security_ops->inode_setsecurity(inode, name, value, size, flags);
+ #endif
+     return -EOPNOTSUPP;
+ }

static inline int security_inode_listsecurity(struct inode *inode, char *buffer, size_t buffer_size)
@@ -1939,47 +1937,47 @@ static inline int security_inode_listsec
#ifdef CONFIG_SECURITY
    if (unlikely (IS_PRIVATE (inode)))
        return 0;
-     return security_ops->inode_listsecurity(inode, buffer, buffer_size);
-#else
-     return 0;
+     if (security_ops->inode_listsecurity)
+         return security_ops->inode_listsecurity(inode, buffer, buffer_size);
+ #endif
+     return 0;
+ }

static inline int security_file_permission (struct file *file, int mask)
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
{
#ifdef CONFIG_SECURITY
-   return security_ops->file_permission (file, mask);
-#else
-   return 0;
+   if (security_ops->file_permission)
+       return security_ops->file_permission(file, mask);
#endif
+   return 0;
}

static inline int security_file_alloc (struct file *file)
{
#ifdef CONFIG_SECURITY
-   return security_ops->file_alloc_security (file);
-#else
-   return 0;
+   if (security_ops->file_alloc_security)
+       return security_ops->file_alloc_security(file);
#endif
+   return 0;
}

static inline void security_file_free (struct file *file)
{
#ifdef CONFIG_SECURITY
-   security_ops->file_free_security (file);
-#else
-   return;
+   if (security_ops->file_free_security)
+       return security_ops->file_free_security(file);
#endif
+   return;
}

static inline int security_file_ioctl (struct file *file, unsigned int cmd,
                                     unsigned long arg)
{
#ifdef CONFIG_SECURITY
-   return security_ops->file_ioctl (file, cmd, arg);
-#else
-   return 0;
+   if (security_ops->file_ioctl)
+       return security_ops->file_ioctl(file, cmd, arg);
#endif
+   return 0;
}

static inline int security_file_mmap (struct file *file, unsigned long reqprot,
@@ -1987,10 +1985,10 @@ static inline int security_file_mmap (st
                                     unsigned long flags)
{
#ifdef CONFIG_SECURITY
-   return security_ops->file_mmap (file, reqprot, prot, flags);
-#else
-   return 0;
+   if (security_ops->file_mmap)
+       return security_ops->file_mmap(file, reqprot, prot, flags);
#endif
+   return 0;
}
```


Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+     return 0;
+ }

static inline int security_file_receive (struct file *file)
{
#ifdef CONFIG_SECURITY
-     return security_ops->file_receive (file);
-#else
-     return 0;
+     if (security_ops->file_receive)
+         return security_ops->file_receive(file);
#endif
+     return 0;
+ }

static inline int security_task_create (unsigned long clone_flags)
{
#ifdef CONFIG_SECURITY
-     return security_ops->task_create (clone_flags);
-#else
-     return 0;
+     if (security_ops->task_create)
+         return security_ops->task_create(clone_flags);
#endif
+     return 0;
+ }

static inline int security_task_alloc (struct task_struct *p)
{
#ifdef CONFIG_SECURITY
-     return security_ops->task_alloc_security (p);
-#else
-     return 0;
+     if (security_ops->task_alloc_security)
+         return security_ops->task_alloc_security(p);
#endif
+     return 0;
+ }

static inline void security_task_free (struct task_struct *p)
{
#ifdef CONFIG_SECURITY
-     security_ops->task_free_security (p);
-#else
-     return;
+     if (security_ops->task_free_security)
+         return security_ops->task_free_security(p);
#endif
+     return;
+ }

static inline int security_task_setuid (uid_t id0, uid_t id1, uid_t id2,
                                       int flags)
{
#ifdef CONFIG_SECURITY
-     return security_ops->task_setuid (id0, id1, id2, flags);
-#else
-     return 0;
+     if (security_ops->task_setuid)
+         return security_ops->task_setuid(id0, id1, id2, flags);
#endif
+     return 0;
+ }
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
}

static inline int security_task_post_setuid (uid_t old_ruid, uid_t old_euid,
                                             uid_t old_suid, int flags)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_post_setuid (old_ruid, old_euid, old_suid, flags);
-#else
-   return cap_task_post_setuid (old_ruid, old_euid, old_suid, flags);
+   if (security_ops->task_post_setuid)
+       return security_ops->task_post_setuid(old_ruid, old_euid, old_suid, flags);
#endif
+   return cap_task_post_setuid (old_ruid, old_euid, old_suid, flags);
}

static inline int security_task_setgid (gid_t id0, gid_t id1, gid_t id2,
                                       int flags)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_setgid (id0, id1, id2, flags);
-#else
-   return 0;
+   if (security_ops->task_setgid)
+       return security_ops->task_setgid(id0, id1, id2, flags);
#endif
+   return 0;
}

static inline int security_task_setpgid (struct task_struct *p, pid_t pgid)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_setpgid (p, pgid);
-#else
-   return 0;
+   if (security_ops->task_setpgid)
+       return security_ops->task_setpgid(p, pgid);
#endif
+   return 0;
}

static inline int security_task_getpgid (struct task_struct *p)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_getpgid (p);
-#else
-   return 0;
+   if (security_ops->task_getpgid)
+       return security_ops->task_getpgid(p);
#endif
+   return 0;
}

static inline int security_task_getsid (struct task_struct *p)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_getsid (p);
-#else
-   return 0;
+   if (security_ops->task_getsid)
+       return security_ops->task_getsid(p);
#endif
+   return 0;
}
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
}

static inline int security_task_setgroups (struct group_info *group_info)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_setgroups (group_info);
-#else
-   return 0;
+   if (security_ops->task_setgroups)
+       return security_ops->task_setgroups(group_info);
#endif
+   return 0;
}

static inline int security_task_setnice (struct task_struct *p, int nice)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_setnice (p, nice);
-#else
-   return 0;
+   if (security_ops->task_setnice)
+       return security_ops->task_setnice(p, nice);
#endif
+   return 0;
}

static inline int security_task_setrlimit (unsigned int resource,
                                          struct rlimit *new_rlim)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_setrlimit (resource, new_rlim);
-#else
-   return 0;
+   if (security_ops->task_setrlimit)
+       return security_ops->task_setrlimit(resource, new_rlim);
#endif
+   return 0;
}

static inline int security_task_setscheduler (struct task_struct *p,
@@ -2169,38 +2167,38 @@ static inline int security_task_setsched
                                          struct sched_param *lp)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_setscheduler (p, policy, lp);
-#else
-   return 0;
+   if (security_ops->task_setscheduler)
+       return security_ops->task_setscheduler(p, policy, lp);
#endif
+   return 0;
}

static inline int security_task_getscheduler (struct task_struct *p)
{
#ifdef CONFIG_SECURITY
-   return security_ops->task_getscheduler (p);
-#else
-   return 0;
+   if (security_ops->task_getscheduler)
+       return security_ops->task_getscheduler(p);
#endif
}
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+     return 0;
+ }

static inline int security_task_kill (struct task_struct *p,
                                     struct siginfo *info, int sig)
{
#ifdef CONFIG_SECURITY
-     return security_ops->task_kill (p, info, sig);
-#else
-     return 0;
+     if (security_ops->task_kill)
+         return security_ops->task_kill(p, info, sig);
#endif
+     return 0;
+ }

static inline int security_task_wait (struct task_struct *p)
{
#ifdef CONFIG_SECURITY
-     return security_ops->task_wait (p);
-#else
-     return 0;
+     if (security_ops->task_wait)
+         return security_ops->task_wait(p);
#endif
+     return 0;
+ }

static inline int security_task_prctl (int option, unsigned long arg2,
@@ -2209,103 +2207,103 @@ static inline int security_task_prctl (i
                                     unsigned long arg5)
{
#ifdef CONFIG_SECURITY
-     return security_ops->task_prctl (option, arg2, arg3, arg4, arg5);
-#else
-     return 0;
+     if (security_ops->task_prctl)
+         return security_ops->task_prctl(option, arg2, arg3, arg4, arg5);
#endif
+     return 0;
+ }

static inline void security_task_reparent_to_init (struct task_struct *p)
{
#ifdef CONFIG_SECURITY
-     security_ops->task_reparent_to_init (p);
-#else
-     cap_task_reparent_to_init (p);
+     if (security_ops->task_reparent_to_init)
+         return security_ops->task_reparent_to_init(p);
#endif
+     cap_task_reparent_to_init (p);
+ }

static inline void security_task_to_inode(struct task_struct *p, struct inode *inode)
{
#ifdef CONFIG_SECURITY
-     security_ops->task_to_inode(p, inode);
-#else
-     return;
+     if (security_ops->task_to_inode)
+         return security_ops->task_to_inode(p, inode);
+ }
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
#endif
+     return;
}

static inline int security_ipc_permission (struct kern_ipc_perm *ipcp,
                                         short flag)
{
#ifdef CONFIG_SECURITY
-     return security_ops->ipc_permission (ipcp, flag);
-#else
-     return 0;
+     if (security_ops->ipc_permission)
+         return security_ops->ipc_permission(ipcp, flag);
#endif
+     return 0;
}

static inline int security_msg_msg_alloc (struct msg_msg * msg)
{
#ifdef CONFIG_SECURITY
-     return security_ops->msg_msg_alloc_security (msg);
-#else
-     return 0;
+     if (security_ops->msg_msg_alloc_security)
+         return security_ops->msg_msg_alloc_security(msg);
#endif
+     return 0;
}

static inline void security_msg_msg_free (struct msg_msg * msg)
{
#ifdef CONFIG_SECURITY
-     security_ops->msg_msg_free_security(msg);
-#else
-     return;
+     if (security_ops->msg_msg_free_security)
+         return security_ops->msg_msg_free_security(msg);
#endif
+     return;
}

static inline int security_msg_queue_alloc (struct msg_queue *msq)
{
#ifdef CONFIG_SECURITY
-     return security_ops->msg_queue_alloc_security (msq);
-#else
-     return 0;
+     if (security_ops->msg_queue_alloc_security)
+         return security_ops->msg_queue_alloc_security(msq);
#endif
+     return 0;
}

static inline void security_msg_queue_free (struct msg_queue *msq)
{
#ifdef CONFIG_SECURITY
-     security_ops->msg_queue_free_security (msq);
-#else
-     return;
+     if (security_ops->msg_queue_free_security)
+         return security_ops->msg_queue_free_security(msq);
#endif
}

```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+     return;
+ }

static inline int security_msg_queue_associate (struct msg_queue * msq,
                                               int msqflg)
{
#ifdef CONFIG_SECURITY
-     return security_ops->msg_queue_associate (msq, msqflg);
-#else
-     return 0;
+     if (security_ops->msg_queue_associate)
+         return security_ops->msg_queue_associate(msq, msqflg);
#endif
+     return 0;
+ }

static inline int security_msg_queue_msgctl (struct msg_queue * msq, int cmd)
{
#ifdef CONFIG_SECURITY
-     return security_ops->msg_queue_msgctl (msq, cmd);
-#else
-     return 0;
+     if (security_ops->msg_queue_msgctl)
+         return security_ops->msg_queue_msgctl(msq, cmd);
#endif
+     return 0;
+ }

static inline int security_msg_queue_msgsnd (struct msg_queue * msq,
                                             struct msg_msg * msg, int msqflg)
{
#ifdef CONFIG_SECURITY
-     return security_ops->msg_queue_msgsnd (msq, msg, msqflg);
-#else
-     return 0;
+     if (security_ops->msg_queue_msgsnd)
+         return security_ops->msg_queue_msgsnd(msq, msg, msqflg);
#endif
+     return 0;
+ }

static inline int security_msg_queue_msgrcv (struct msg_queue * msq,
@@ -2314,93 +2312,93 @@ static inline int security_msg_queue_msg
                                             long type, int mode)
{
#ifdef CONFIG_SECURITY
-     return security_ops->msg_queue_msgrcv (msq, msg, target, type, mode);
-#else
-     return 0;
+     if (security_ops->msg_queue_msgrcv)
+         return security_ops->msg_queue_msgrcv(msq, msg, target, type, mode);
#endif
+     return 0;
+ }

static inline int security_shm_alloc (struct shmid_kernel *shp)
{
#ifdef CONFIG_SECURITY
-     return security_ops->shm_alloc_security (shp);
-#else
-     return 0;
+     if (security_ops->shm_alloc_security)
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+         return security_ops->shm_alloc_security(shp);
+     #endif
+     return 0;
+ }

static inline void security_shm_free (struct shmid_kernel *shp)
{
+ #ifdef CONFIG_SECURITY
-     security_ops->shm_free_security (shp);
- #else
-     return;
+     if (security_ops->shm_free_security)
+         return security_ops->shm_free_security(shp);
+ #endif
+     return;
+ }

static inline int security_shm_associate (struct shmid_kernel * shp,
                                         int shmflg)
{
+ #ifdef CONFIG_SECURITY
-     return security_ops->shm_associate(shp, shmflg);
- #else
-     return 0;
+     if (security_ops->shm_associate)
+         return security_ops->shm_associate(shp, shmflg);
+ #endif
+     return 0;
+ }

static inline int security_shm_shmctl (struct shmid_kernel * shp, int cmd)
{
+ #ifdef CONFIG_SECURITY
-     return security_ops->shm_shmctl (shp, cmd);
- #else
-     return 0;
+     if (security_ops->shm_shmctl)
+         return security_ops->shm_shmctl(shp, cmd);
+ #endif
+     return 0;
+ }

static inline int security_shm_shmat (struct shmid_kernel * shp,
                                     char __user *shmaddr, int shmflg)
{
+ #ifdef CONFIG_SECURITY
-     return security_ops->shm_shmat(shp, shmaddr, shmflg);
- #else
-     return 0;
+     if (security_ops->shm_shmat)
+         return security_ops->shm_shmat(shp, shmaddr, shmflg);
+ #endif
+     return 0;
+ }

static inline int security_sem_alloc (struct sem_array *sma)
{
+ #ifdef CONFIG_SECURITY
-     return security_ops->sem_alloc_security (sma);
- #else
-     return 0;
+     if (security_ops->sem_alloc_security)
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+         return security_ops->sem_alloc_security(sma);
+     #endif
+     return 0;
+ }

static inline void security_sem_free (struct sem_array *sma)
{
+ #ifdef CONFIG_SECURITY
-     security_ops->sem_free_security (sma);
- #else
-     return;
+     if (security_ops->sem_free_security)
+         return security_ops->sem_free_security(sma);
+ #endif
+     return;
+ }

static inline int security_sem_associate (struct sem_array * sma, int semflg)
{
+ #ifdef CONFIG_SECURITY
-     return security_ops->sem_associate (sma, semflg);
- #else
-     return 0;
+     if (security_ops->sem_associate)
+         return security_ops->sem_associate(sma, semflg);
+ #endif
+     return 0;
+ }

static inline int security_sem_semctl (struct sem_array * sma, int cmd)
{
+ #ifdef CONFIG_SECURITY
-     return security_ops->sem_semctl(sma, cmd);
- #else
-     return 0;
+     if (security_ops->sem_semctl)
+         return security_ops->sem_semctl(sma, cmd);
+ #endif
+     return 0;
+ }

static inline int security_sem_semop (struct sem_array * sma,
@@ -2408,10 +2406,10 @@ static inline int security_sem_semop (st
int alter)
{
+ #ifdef CONFIG_SECURITY
-     return security_ops->sem_semop(sma, sops, nsops, alter);
- #else
-     return 0;
+     if (security_ops->sem_semop)
+         return security_ops->sem_semop(sma, sops, nsops, alter);
+ #endif
+     return 0;
+ }

static inline void security_d_instantiate (struct dentry *dentry, struct inode *inode)
@@ -2419,46 +2417,46 @@ static inline void security_d_instantiat
+ #ifdef CONFIG_SECURITY
+     if (unlikely (inode && IS_PRIVATE (inode)))
+         return;
-     security_ops->d_instantiate (dentry, inode);
- #else
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
-     return;
+     if (security_ops->d_instantiate)
+         return security_ops->d_instantiate(dentry, inode);
#endif
+     return;
}

static inline int security_getprocattr(struct task_struct *p, char *name, void *value, size_t size)
{
#ifdef CONFIG_SECURITY
-     return security_ops->getprocattr(p, name, value, size);
-#else
-     return -EINVAL;
+     if (security_ops->getprocattr)
+         return security_ops->getprocattr(p, name, value, size);
#endif
+     return -EINVAL;
}

static inline int security_setprocattr(struct task_struct *p, char *name, void *value, size_t size)
{
#ifdef CONFIG_SECURITY
-     return security_ops->setprocattr(p, name, value, size);
-#else
-     return -EINVAL;
+     if (security_ops->setprocattr)
+         return security_ops->setprocattr(p, name, value, size);
#endif
+     return -EINVAL;
}

static inline int security_netlink_send(struct sock *sk, struct sk_buff * skb)
{
#ifdef CONFIG_SECURITY
-     return security_ops->netlink_send(sk, skb);
-#else
-     return cap_netlink_send (sk, skb);
+     if (security_ops->netlink_send)
+         return security_ops->netlink_send(sk, skb);
#endif
+     return cap_netlink_send (sk, skb);
}

static inline int security_netlink_recv(struct sk_buff * skb)
{
#ifdef CONFIG_SECURITY
-     return security_ops->netlink_recv(skb);
-#else
-     return cap_netlink_recv (skb);
+     if (security_ops->netlink_recv)
+         return security_ops->netlink_recv(skb);
#endif
+     return cap_netlink_recv (skb);
}

static inline int security_unix_stream_connect(struct socket * sock,
@@ -2466,10 +2464,10 @@ static inline int security_unix_stream_c
                                struct sock * newsk)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->unix_stream_connect(sock, other, newsk);
-#else
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
-     return 0;
+     if (security_ops->unix_stream_connect)
+         return security_ops->unix_stream_connect(sock, other, newsk);
#endif
+     return 0;
}

@@ -2477,20 +2475,20 @@ static inline int security_unix_may_send
                                struct socket * other)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->unix_may_send(sock, other);
-#else
-     return 0;
+     if (security_ops->unix_may_send)
+         return security_ops->unix_may_send(sock, other);
#endif
+     return 0;
}

static inline int security_socket_create (int family, int type,
                                int protocol, int kern)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_create(family, type, protocol, kern);
-#else
-     return 0;
+     if (security_ops->socket_create)
+         return security_ops->socket_create(family, type, protocol, kern);
#endif
+     return 0;
}

static inline void security_socket_post_create(struct socket * sock,
@@ -2499,11 +2497,10 @@ static inline void security_socket_post_
                                int protocol, int kern)
{
#ifdef CONFIG_SECURITY_NETWORK
-     security_ops->socket_post_create(sock, family, type,
-                                     protocol, kern);
-#else
-     return;
+     if (security_ops->socket_post_create)
+         return security_ops->socket_post_create(sock, family, type, protocol, kern);
#endif
+     return;
}

static inline int security_socket_bind(struct socket * sock,
@@ -2511,10 +2508,10 @@ static inline int security_socket_bind(s
                                int addrlen)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_bind(sock, address, addrlen);
-#else
-     return 0;
+     if (security_ops->socket_bind)
+         return security_ops->socket_bind(sock, address, addrlen);
#endif
+     return 0;
}
```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
static inline int security_socket_connect(struct socket * sock,
@@ -2522,49 +2519,49 @@ static inline int security_socket_connec
                                int addrlen)
{
#ifdef CONFIG_SECURITY_NETWORK
-   return security_ops->socket_connect(sock, address, addrlen);
-#else
-   return 0;
+   if (security_ops->socket_connect)
+       return security_ops->socket_connect(sock, address, addrlen);
#endif
+   return 0;
}

static inline int security_socket_listen(struct socket * sock, int backlog)
{
#ifdef CONFIG_SECURITY_NETWORK
-   return security_ops->socket_listen(sock, backlog);
-#else
-   return 0;
+   if (security_ops->socket_listen)
+       return security_ops->socket_listen(sock, backlog);
#endif
+   return 0;
}

static inline int security_socket_accept(struct socket * sock,
                                        struct socket * newsock)
{
#ifdef CONFIG_SECURITY_NETWORK
-   return security_ops->socket_accept(sock, newsock);
-#else
-   return 0;
+   if (security_ops->socket_accept)
+       return security_ops->socket_accept(sock, newsock);
#endif
+   return 0;
}

static inline void security_socket_post_accept(struct socket * sock,
                                              struct socket * newsock)
{
#ifdef CONFIG_SECURITY_NETWORK
-   security_ops->socket_post_accept(sock, newsock);
-#else
-   return;
+   if (security_ops->socket_post_accept)
+       return security_ops->socket_post_accept(sock, newsock);
#endif
+   return;
}

static inline int security_socket_sendmsg(struct socket * sock,
                                        struct msghdr * msg, int size)
{
#ifdef CONFIG_SECURITY_NETWORK
-   return security_ops->socket_sendmsg(sock, msg, size);
-#else
-   return 0;
+   if (security_ops->socket_sendmsg)
+       return security_ops->socket_sendmsg(sock, msg, size);

```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
#endif
+     return 0;
}

static inline int security_socket_recvmsg(struct socket * sock,
@@ -2572,95 +2569,95 @@ static inline int security_socket_recvmsg
        int flags)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_recvmsg(sock, msg, size, flags);
-#else
-     return 0;
+     if (security_ops->socket_recvmsg)
+         return security_ops->socket_recvmsg(sock, msg, size, flags);
#endif
+     return 0;
}

static inline int security_socket_getsockname(struct socket * sock)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_getsockname(sock);
-#else
-     return 0;
+     if (security_ops->socket_getsockname)
+         return security_ops->socket_getsockname(sock);
#endif
+     return 0;
}

static inline int security_socket_getpeername(struct socket * sock)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_getpeername(sock);
-#else
-     return 0;
+     if (security_ops->socket_getpeername)
+         return security_ops->socket_getpeername(sock);
#endif
+     return 0;
}

static inline int security_socket_getsockopt(struct socket * sock,
        int level, int optname)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_getsockopt(sock, level, optname);
-#else
-     return 0;
+     if (security_ops->socket_getsockopt)
+         return security_ops->socket_getsockopt(sock, level, optname);
#endif
+     return 0;
}

static inline int security_socket_setsockopt(struct socket * sock,
        int level, int optname)
{
#ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_setsockopt(sock, level, optname);
-#else
-     return 0;

```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+     if (security_ops->socket_setsockopt)
+         return security_ops->socket_setsockopt(sock, level, optname);
+ #endif
+     return 0;
+ }

static inline int security_socket_shutdown(struct socket * sock, int how)
{
+ #ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_shutdown(sock, how);
- #else
-     return 0;
+     if (security_ops->socket_shutdown)
+         return security_ops->socket_shutdown(sock, how);
+ #endif
+     return 0;
+ }

static inline int security_sock_rcv_skb (struct sock * sk,
                                         struct sk_buff * skb)
{
+ #ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_sock_rcv_skb (sk, skb);
- #else
-     return 0;
+     if (security_ops->socket_sock_rcv_skb)
+         return security_ops->socket_sock_rcv_skb(sk, skb);
+ #endif
+     return 0;
+ }

static inline int security_socket_getpeersec(struct socket *sock, char __user *optval,
                                             int __user *optlen, unsigned len)
{
+ #ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->socket_getpeersec(sock, optval, optlen, len);
- #else
-     return -ENOPROTOOPT;
+     if (security_ops->socket_getpeersec)
+         return security_ops->socket_getpeersec(sock, optval, optlen, len);
+ #endif
+     return -ENOPROTOOPT;
+ }

static inline int security_sk_alloc(struct sock *sk, int family, int priority)
{
+ #ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->sk_alloc_security(sk, family, priority);
- #else
-     return 0;
+     if (security_ops->sk_alloc_security)
+         return security_ops->sk_alloc_security(sk, family, priority);
+ #endif
+     return 0;
+ }

static inline void security_sk_free(struct sock *sk)
{
+ #ifdef CONFIG_SECURITY_NETWORK
-     return security_ops->sk_free_security(sk);
- #else
-     return;
+ #endif
+ }

```

Linux-Kernel: [PATCH 3/5] Call security hooks conditionally if the security_op is filled out.

```
+     if (security_ops->sk_free_security)
+         return security_ops->sk_free_security(sk);
+ #endif
+     return;
+ }
```

```
#endif /* __LINUX_SECURITY_H */
```

--

-

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>